Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

Ontology-Based Information Extraction with a Cognitive Agent

Peter Lindes, Deryle W. Lonsdale, David W. Embley
Brigham Young University
Provo, Utah 84602

Abstract

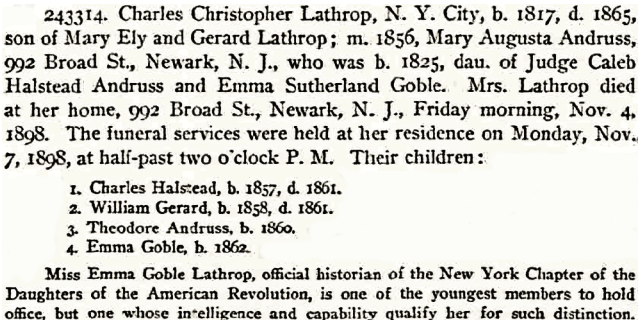
Machine reading is a relatively new field that features computer programs designed to read flowing text and extract fact assertions expressed by the narrative content. This task involves two core technologies: natural language processing (NLP) and information extraction (IE). In this paper we describe a machine reading system that we have developed within a cognitive architecture. We show how we have integrated into the framework several levels of knowledge for a particular domain, ideas from cognitive semantics and construction grammar, plus tools from prior NLP and IE research. The result is a system that is capable of reading and interpreting complex and fairly idiosyncratic texts in the family history domain. We describe the architecture and performance of the system. After presenting the results from several evaluations that we have carried out, we summarize possible future directions.

Introduction

Much Web traffic involves people searching for genealogical data that might inform them about their family history. A great online supply of historical documents containing such data exists, but most were generated long before modern digital technology was available. In this paper we discuss one way of extracting information from historical documents in a digital form so it can be searchable.

Various approaches are possible depending upon the type of document involved. Census records are highly structured but are largely handwritten. Involving vast pools of human annotators to hand-index entries has proven useful in this case, as was shown recently for the 1940 U.S. census.

A large corpus of family history books written before the digital age is now also becoming available online. Figure 1 shows a short example of text from p. 419 of one such book (Vanderpoel 1902). Many parts of this book have information in a fairly structured form, as can be seen in the list of children. However, much of the rest of the text follows a greatly abbreviated and highly formulaic style of English lexis and grammar. Typically these books were digitized by scanning them into PDF files and using optical character recognition (OCR) algorithms to extract the raw text. Of course the OCR process introduces a sizable number of errors. Once a book has been digitized, manual annotation and indexation can be performed to identify interesting facts using existing tools. However, the task is enormous, challenging, and complicated.



243314. Charles Christopher Lathrop, N. Y. City, b. 1817, d. 1865, son of Mary Ely and Gerard Lathrop; m. 1856, Mary Augusta Andruss, 992 Broad St., Newark, N. J., who was b. 1825, dau. of Judge Caleb Halstead Andruss and Emma Sutherland Goble. Mrs. Lathrop died at her home, 992 Broad St., Newark, N. J., Friday morning, Nov. 4, 1898. The funeral services were held at her residence on Monday, Nov. 7, 1898, at half-past two o'clock P. M. Their children:

1. Charles Halstead, b. 1857, d. 1861.
2. William Gerard, b. 1858, d. 1861.
3. Theodore Andruss, b. 1860.
4. Emma Goble, b. 1862.

Miss Emma Goble Lathrop, official historian of the New York Chapter of the Daughters of the American Revolution, is one of the youngest members to hold office, but one whose intelligence and capability qualify her for such distinction.

Figure 1: Sample of domain text (Vanderpoel 1902).

Another solution is to automatically extract information from the textual content. This requires integrating several types and levels of knowledge: lexical, syntactic, semantic, and pragmatic. In this paper we describe a system called OntoSoar that is designed to apply all these levels of knowledge to the problem of extracting information from genealogy books in an automatic fashion. Built within a cognitive architecture framework and targeting information specified by a user-supplied ontology, its advanced text processing and information extraction functions integrate seamlessly. We discuss performance of the system and evaluate it against a gold-standard corpus of human annotations.

Related work

Our work derives from several principal threads of ongoing research: natural language processing (NLP), information extraction, cognitive grammar, and deep reading.

The task is largely linguistic since we are processing running text. However, as mentioned above, the text departs from normal expectations of "grammatical" English; this complicates the syntactic processing—or parsing—stage.

We (Lonsdale et al. 2007) and others (Akbik and Bross 2009) have successfully used the link grammar (LG) parser to parse text that, like ours, is linguistically idiosyncratic and

varies from book to book. The LG parser is an open-source parsing tool that is both robust and flexible (Sleator and Temperley 1993). It looks for pairwise associations (linkages) between words and annotates the linkages with labels and (optionally) scores.

Our text processing, though, must go beyond mere syntactic parsing: we require further semantic treatment to recognize the concepts and relationships of interest. Some (but not all) work in semantic formalisms has sought to account for human language processing and the deep structures needed to understand literal and non-literal meaning (Jackendoff 1990; 1996; 2002; 2003; Lakoff and Johnson 1980) including grounding it in direct perceptual experience, agency, and embodiment (Johnson 1987). This area of cognitive linguistics is finding increasing application in systems for understanding human language.

One thread of cognitive semantic systems research has resulted in a concrete language processing system (Feldman 2006; Bryant 2008; Chang 2008). A central component of this research is a grammatical theory called Embodied Construction Grammar (Bergen and Chang 2013). Hoffman and Trousdale (2013) survey common construction grammar tenets shared across approaches, which are directly relevant to our current implementation.

Another branch of computer science has tried to build functioning models of human cognition, called cognitive architectures. These theories and the associated implementations draw heavily on experimental evidence from psychology and measurements of how the brain processes information. Anderson (2007) gives a good introduction to this field. Soar is a mature and versatile cognitive architecture (Newell 1990; Laird 2012) that has been applied to many application areas. We use Soar as our framework for representing meaning and performing reasoning on it across several complex knowledge structures. Cognitive modeling of language use in Soar was pioneered by the NL-Soar system (Lewis 1993), which parses sentences using methods inspired by psycholinguistic research on human sentence processing.

We have built on the core Soar cognitive architecture, finding it to be a good candidate to act as an agent for language understanding. LG-Soar is a Soar-based language processing system that uses the LG parser discussed above, along with a semantic interpreter developed inside Soar to extract meaning from input sentences. LG-Soar has been used for information extraction applications (Lonsdale et al. 2008) and in a robotics system that can learn new linguistic constructions (Mohan et al. 2012). The present work derives from this approach, but with an innovative form of semantic analyzer. Soar's use in this project derives in part from the fact that Soar is intended to model human cognition (Newell 1990) and by the importance of agency in understanding language (Melby and Warner 1995).

Several and varied methods exist for extracting useful information from the wide range of existing text types. Sarawagi (2007) reviews the whole field of information extraction. Buitelaar et al. (2009) present an approach to linguistic grounding of ontologies. They argue that "currently available data-models are not sufficient . . . without linguistic grounding or structure". Notably absent from these discussions is consideration of a deep understanding of language.

Another approach called "machine reading" is discussed in depth by (Hruschka 2013). He reviews three systems for building knowledge bases by machine reading the web: YAGO, KnowItAll, and NELL. Each system starts with some seed knowledge and uses various techniques to make both the accumulated set of fact assertions and the underlying ontology grow by reading large amounts of knowledge from the web. However, these systems still have fairly low accuracy in extracting individual facts and are not tuned to the special sublanguages of English such as those used in many family history books.

One of the features of OntoSoar is its ability to take extracted information in its internal representation of the meaning of input text and transform that information to populate an ontology[1] provided by the user. This amounts to a special case of the general problem of ontology matching, for which there is also a large literature.

An overview and survey of this field is given by Euzenat and Schvaiko (2007) . Bleiholder and Naumann (2008) and Mitra, Noy, and Jaizwal (2004), as well as many others, discuss specific approaches in more detail. Most of this literature deals with how to map information from one web site to another, or onto some pre-defined ontology. Fortunately for us our ontology mapping problem is much simpler since we are working within a well understood domain.

Our work unites these various streams of computational research and illustrates how the result can process certain types of data-rich text. Since our system uses both ontology-based and Soar-based processing, we call it OntoSoar. The innovative semantic analyzer described here is based on a number of ideas derived from the literature on cognitive semantics, construction grammar, and cognitive architectures.

For ontology specification we use the OntoES data extraction approach (Embley, Liddle, and Lonsdale 2011) and conceptual modeling system (Embley, Kurtz, and Woodfield 1992). In part OntoES draws on a large body of literature on conceptual modeling to produce a framework called OSM capable of representing a wide variety of conceptual models and populating them with data.

Figure 2 shows a sample user ontology. The ontology object and relationship sets are based on simple English sentences and phrases. Reading via arrow direction yields full names for relationship sets: e.g. "Person born on BirthDate", "Person died on DeathDate", "Person married Spouse on MarriageDate", "Son of Person", "Daughter of Person". OntoSoar uses these names for matching Soar conceptualizations with ontology conceptualizations.

OSM is a logic-based representation: the object sets are 1-place predicates, the n-ary relations are n-place predicates, and the constraints are representable as well-formed formulas. Whereas the figure presents the ontology in its graphical illustration form, its contents could also be listed entirely in the predicate logic formulation. Our OntoES system can translate extracted information directly into RDF and OWL.

[1]We use the term "ontology" as usually found in information extraction literature: a computerized conceptual model that can be populated with facts (Gruber 1993).

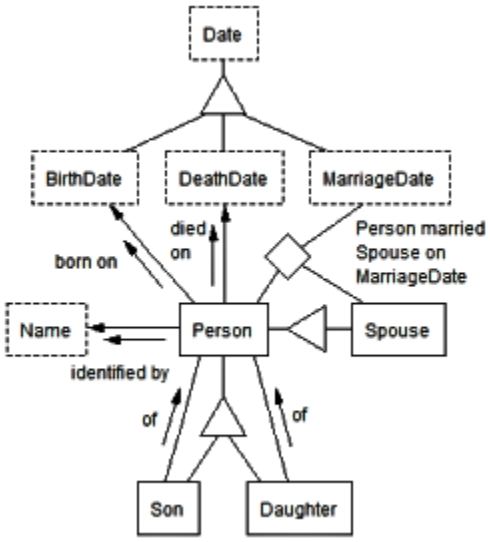


Figure 2: Sample ontology.

OntoSoar fits into this overall OntoES system by reading in a user ontology in OSMX[2] form and outputting a modified OSMX file which contains the fact assertions it found in a given input text. In addition, OntoSoar can be evaluated by using the OntoES tools to compare the fact assertions found by OntoSoar with those found by a human annotator in the same text.

OntoSoar processing

In this section we sketch how we use lexical, syntactic, and semantic analysis tools to extract information from data-rich texts and match that data to a conceptual model of the family history domain provided by a user, populating that model with fact assertions found in the text. Several levels of knowledge interact in our system: lexical, syntactic, semantic, and pragmatic.

OntoSoar is built using Java components, some Java libraries, some custom Java components, the LG parser, the Soar system, and Soar code that implements all the semantic components. Figure 3 shows a basic overview of the processing pipeline, with the superordinate arrow labeled "Soar" indicating the cognitive agent framework.

To summarize, each book is digitized via camera capture, and the images undergo optical character recognition (OCR) analysis. The resulting PDF files serve as input to OntoSoar, along with an OSMX target ontology that specifies the targeted content. OntoSoar divides the raw input text into sentence-like segments, then processes the pages one segment at a time, parsing each segment and performing semantic interpretation on the result. A mapping component takes a conceptual model in the form of an OSMX file, populates it with fact assertions derived from the semantic structures, and outputs the populated ontology as a new OSMX file. This output file can then be viewed, evaluated, or imported into a database by tools from the OntoES tool set.

[2]OSMX is the XML file format we use for storing conceptual ontologies and their content in the Object-oriented Systems Modeling (OSM) conceptual modeling language.

Linguistic processing

OntoSoar first divides the raw input text into segments corresponding to short sentences. Tokenization includes word-level corrections to reduce OCR errors and relexing tokens acting as abbreviations in the domain: born for b., died for d., daughter for dau., and so on. Often, to avoid repetition, pronouns (he, she, they, etc.) are elided from genealogical text; in such cases we insert a temporary placeholder token GP (Generic Pronoun).

The next step involves parsing the incoming segment with the link grammar (LG) parser. Consider this partial example linkage produced for the sample text:

```
2: Charles Christopher Lathrop, N. Y. City, born 1817, died 1865,
   son of Mary Ely and Gerard Lathrop ;  ';'


                      +-----------------Ss-------------
                      +------MX------+-------Xc-------+
                      |    +----Xd---+--MX*p-+---Xca--+
   +----G----+----G----+    |  +-G+-G-+  +-Xd-+--IN-+  |
   |         |         |    |  |  |   |  |  |     |  |  |
Charles Christopher Lathrop , N. Y. City , born.v 1817 ,
```

G links build proper nouns, and the Ss link connects the subject with the verb (not illustrated). Careful inspection shows two incorrect MX (i.e. appositive) links, indicating that the LG parser thinks that "Charles Christopher Lathrop" is "N. Y. City" and that the latter was born in 1817. However, this is corrected downstream by the semantic processor.

The Meaning Builder is a component based on Embodied Construction Grammar (ECG) (Bergen and Chang 2013; Bryant 2008; Chang 2008), though we depart from the core theory in two fundamental ways. First, construction grammar in general and ECG in particular build constructions directly from input text. However, OntoSoar builds constructions from the LG parser linkages, allowing OntoSoar to act on information about words and their relationships.

(Bryant 2008) uses a compiler for converting a formal grammar written in this ECG language into an internal form; we instead hand-coded ECG rules as Soar productions. Some of these are declarative, building static data structures. Other productions fire as the semantic analysis is proceeding and thus enact procedural knowledge. In the future we may be able to build a compiler to convert ECG grammar rules directly into Soar code.

Figure 4 shows part of the LG linkage for our sample segment and a set of overarching rectangles and arrows that represent the constructions recognized from this segment. The lower level rectangles have arrows pointing to the words that make up the form pole (i.e. anchors) of each of those constructions. Though not shown, each construction builds on words that it contains and on the leftward links from each word. The ovals represent meaning structures built from the constructions.

LifeEvent structures form the root of meaning networks. Each meaning structure has a number of internal slots—called roles—(not shown) that store values of properties or references to other meaning structures. For example,

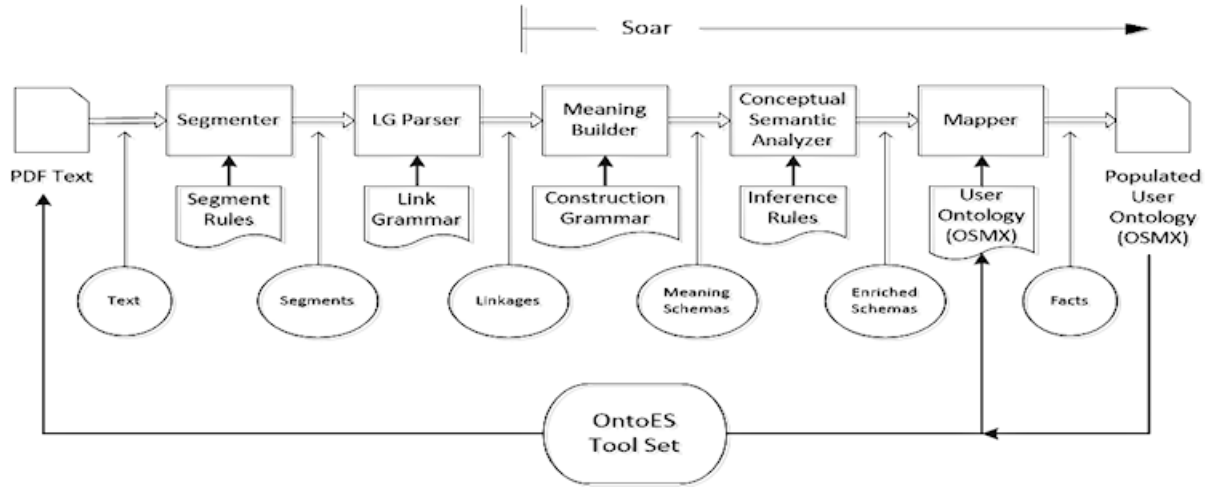


Figure 3: OntoSoar system architecture

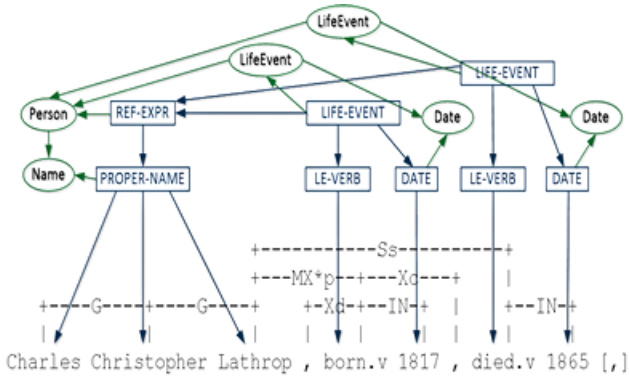


Figure 4: Example of construction with meanings

a LifeEvent has a subject role requiring a Person and a date role requiring a Date. A Person's name role takes a Name, but it also has birth and death roles which point to LifeEvents, if filled.

For each input word OntoSoar executes a comprehend-word operator that in turn invokes a build-meaning operator. Other operators such as lexical-construction and proper-name perform specialized word-based mappings for building these semantic structures.

The Meaning Builder expects arguments of type Person for predicates such as "was born" or "son of", hence a proper name refers to a Person if it is an argument of such a predicate. In this way the Meaning Builder constructs a network of meaning structures with their roles, many of which are not yet filled. This network provides the basis for further semantic analysis.

The Conceptual Semantic Analyzer takes the meaning structures supplied by the Meaning Builder and expands and enhances them using inference rules implemented as Soar productions. For example, the presence of a phrase like "is not living" in the text triggers search for a death event for the subject person, and also a Date schema with its value set to UNKNOWN; when a death date is reported, its value is set explicitly. Similarly, when the text mentions that someone "is married" without further comment, the system will infer the existence of a second person whose name is UNKNOWN. Another example of inference involves reference resolution, where pronouns and GP tokens are associated by backward search for the Person they refer to. This is implemented with the resolve-reference operator, which performs basic search but does not yet take advantage of gender and number agreement or the specific meanings of nouns like "widow".

In OntoSoar the meaning schemas are modeled on image schemas (Johnson 1987), though they are not connected to perception in any direct way. Nevertheless the matching of one schema with roles that connect to other schemas in a network provides declarative knowledge that enables adding the procedural knowledge for inferencing.

Matching target ontologies

The information OntoSoar targets is subject to some simplification: while identifying unique individuals and associated names, gender, important dates, and family relationships, we exclude for now geographical locations plus other facts like employment or religion. We also adopt simplifying assumptions about family relationships, namely that a marriage is between a man and a woman, and that parent/child relationships are only for biological parents. These limitations and assumptions can be relaxed in future work.

Once we have analyzed an input segment to build our internal meaning structures, the final step is to project those meanings onto the ontology provided by the user. This work is done in two steps. Since both the internal meaning schemas of ECG and the user ontology are static, we can find object and relationship sets in the ontology that match parts of our schemas statically before we have seen any input data. Then when a segment has been completely analyzed, we can use these matches to map the specific meanings found in the

segment onto fact assertions in the ontology.

OntoSoar's find-matches operator executes the matching procedure, along with various other operators that match keywords from the internal schemas with words taken from the names of the sets in the ontology:

- A lexical schema matches against any lexical object set (i.e. boxes in Figure 2 with dashed borders) that has a word in its name[3] matching one of the schema keywords.
- The Person schema matches to any object set regardless of its name as long as it has a relationship set connecting to a lexical object set that matches ProperName.
- The Couple schema matches against a pattern with a relationship set with three or more arguments connecting the object set that matches Person with one of its specializations (pointed to by the triangle in Figure 2) and a third argument that matches Date if that relationship set also has married in its name.
- The FamilyRelationship schema has a matching algorithm that looks for specializations of the object set which matches Person whose names contain the keywords son, daughter, or child.
- The LifeEvent schema looks for matches to relationship sets where the name of the relationship set has a word that matches one of the verbs that can generate a LifeEvent (e.g. "born" or "died"). These matches are recorded according to the verb that matches, so that the general LifeEvent schema will match several relationship sets, with the correct match being chosen later on according to the specific verb present. This matching also connects to the correct specialization of Person.

When the semantic analysis of a given segment has been completed, the OntoSoar extract-facts operator projects as many fact assertions as possible from the meanings found for the segment into the user ontology. Separate sub-operators extract assertions according to the various types of matches found previously. This assertion extraction process is fairly straightforward since we have already done the hard part in the matching.

Evaluation and Results

So far we have run several evaluations of OntoSoar's performance, across different documents and while using different user ontologies. First, we processed the sample text page, plus another page relatively similar to it from another book, using the ontology in Figure 2. For each sample text an output OSMX file was produced which contained fact assertions populating the ontology with persons identified by names, birth and death dates, and marriages.

Table 1 shows combined precision, recall, and F-measure result set for Samples 1 and 2 when compared to human annotations. Overall the precision is quite high, but the recall is lower. The primary reason for the recall errors is the lack of understanding of all the linguistic constructions used in the text.

[3]Ontology names are in camel-case and are split by OntoSoar.

Category	Exist	Found	Good	P %	R %	F %
Persons	31	26	25	96.2	80.6	87.7
Births	14	14	13	92.9	92.9	92.9
Deaths	9	7	7	100.0	77.8	87.5
Marriages	7	7	5	71.4	71.4	71.4
Children	16	2	2	100.0	12.5	22.2
Tot./Avg.	77	56	52	92.9	67.5	78.2

Table 1: Combined accuracy measures for Samples 1 and 2

Persons: The system has an ontological committment for creating a Person: there must be a proper name, and that name must be the grammatical subject or object of a predicate which applies to people, such as "born", "married", or "son of".

For the two sample text pages, OntoSoar missed finding six people. One was missed in the first sample text because the last sentence contained no identifiable predicates associated with her mention. In another case in the second text page, a person was incorrectly identified because of a segmentation ambiguity. Another was missed because of OntoSoar's current inability to unpack the dense semantics in the expression "...by whom she had one son...". Other reference resolution problems explain the other missing Persons.

Births and Deaths: OntoSoar finds every birth event, but in one case it is assigned to the wrong person due to unclear reference. Some dates are marked as UNKNOWN when the English text states that a person died (e.g. with the phrase "... is not living") but does not specify the date.

Marriages: OntoSoar finds all mentioned marriages (one in the first sample page, and six in the second), but in two cases from the latter it attaches the wrong subject to them.

Sons and Daughters: Many of the parent-child relationships in these sample texts, and in many other texts as well, are represented as enumerated lists of children. OntoSoar does not yet implement any list processing. This caused a total of 12 recall errors.

Beyond the two sample pages mentioned above, we also evaluated OntoSoar performance on a larger sampling of texts from family history books. We have access to a private repository of over a hundred thousand such books. We selected 200 books at random from this collection, and then randomly chose a sequence of three consecutive and data-rich pages from each of these books. We then arbitrarily chose twelve books' three-page ranges and ran them through OntoSoar. With minor adaptations to the input process, all twelve of the text files ran successfully through OntoSoar.

Performing a complete measure of the precision and recall of OntoSoar on this data would require manually annotating all the texts for all the relations of interest, which was beyond the scope of the available resources. However, we have looked through all the output files to examine the fact assertions that OntoSoar claims to have found and evaluated each as correct or not. The results are summarized in Table 2; the "Ely" and "Myra" files are the two sample pages discussed above.

In evaluating the matches, persons were considered correct if OntoSoar found at least a subset of the name given in the text with no extraneous material. Births and deaths were

		Persons		Births & Deaths		Marriages		Children		Run Time	
File	Segs	Found	Correct	Found	Correct	Found	Correct	Found	Correct	Secs	Segs/Sec
Ely	15	11	100.00%	10	100.00%	1	100.00%	2	100.00%	15	1.000
Myra	23	15	93.33%	11	100.00%	6	66.67%	0	0.00%	10	2.300
Other documents	1547	328	73.48%	176	40.34%	78	51.28%	31	77.42%	1489	1.039

Table 2: Precision results for additional texts

considered correct if they were attached to a legitimate person and the date was complete. A marriage was considered correct if it connected the two correct people, even if the date was not found or incomplete. A child was considered correct if a person of the right gender was connected as a son or daughter to at least one of the correct parents.

The table only gives an estimate of precision; no attempt was made to measure either recall or F-measure. Unsurprisingly, the overall recall for these twelve files is rather low. If no facts were found in a particular case, the precision is marked as N/A. Overall OntoSoar processing time is consistent, at about one second per segment.

Many issues contribute to both recall and precision being much lower than for our original two samples. Some, such as OCR errors, are mostly beyond the reach of OntoSoar to solve. Other types of errors, however, could be reduced substantially by further improvements to OntoSoar within the scope of its existing architecture. For example, OntoSoar currently misses the many instances of dates formatted like "25 June 1823" or "6/25/1823" or even "Private" (indicating a person alive at time of publication). Nor can OntoSoar correctly unpack combined personal name constructions like "John Phillip and Alice Adel (Billeter) Harris", a common structure in this domain. Extending OntoSoar's capabilities in such areas will be straightforward.

Our work is preliminary, so we include a few notes about portability. We have directed very little effort at this specific text type: only the semantic interpretation rules that the agent executes were hand-crafted for the specific text type of this domain. Other text domains would require more semantic rules, but their integration into the system would be straightforward.

OntoSoar's conceptual-model-based ontologies are built by end-users, so presumably they may have different ways of conceptualizing or expressing the desired relationships than in the ontology depicted in Figure 2. We tested OntoSoar performance on two other ontologies representing largely the same information but structured differently. OntoSoar did well in finding Persons, Births, Deaths, and Marriages. However, the second ontology only specified Parent-Child relations, but not with specializations of Child for Son and Daughter, so OntoSoar was not able to distinguish these when using that ontology. The third ontology did not have any object sets for Child, so none of these were recognized. With more inference rules for reasoning about all possible arrangements of family relationships, these connections should be made.

Finally, we ran the system on all 830 pages of the book mentioned earlier (Vanderpoel 1902), which contains the partial page shown in Figure 1. Processing took about 8 hours on a typical PC desktop. The collection of assertions extracted consists of:

Persons	16,848
Births	8,609
Deaths	2,406
Genders	1,674
Couples	3,343
Children	3,049
Total	35,929

Conclusions and Future Work

From our work—preliminary though it is at this point— we are already able to draw several conclusions:

- Our linguistic analysis components are capable of extracting fact assertions from complex genealogical texts. In particular, an agent-based cognitive construction grammar framework provides a viable semantic representation.
- Meaning structures can be mapped onto ontologies to populate a user-specified conceptual model.
- The Soar cognitive architecture supports the above processes and provides a basis for more extensive inferencing for higher-level linguistic issues.

This work also represents a sizable increase in the quantity of running text that any Soar-based system has processed; treating a whole book of hundreds of pages with Soar is unprecedented.

Incremental improvement to OntoSoar is possible by adding or modifying rules in several parts of the system: the Segmenter can be made to recognize and expand new abbreviations such as "b", "dau", and "Bapt."; the grammar of the LG Parser can be augmented to understand different date formats; and the constructions in the Semantic Analyzer can be expanded to recognize common phrases like "his widow" and "they had one son".

Previous work has demonstrated incremental word-by-word language modeling in Soar with parsers other than the LG parser (Rytting and Lonsdale 2006); this permits interleaving syntactic and semantic processing at the word level. The basic pipeline described in this paper could be improved to allow for more interaction between semantics and the parser, the syntax, or even the segmenter.

Several Soar systems have learned tasks by observing humans doing the tasks. We anticipate being able to integrate this ability into our work, since the OntoES Annotator already provides a framework for human annotators doing the same task as OntoSoar. As the system scales up to handle more books of wider coverage, this type of human input would be helpful in the initial stages of processing each new book to adapt to its linguistic style.

References

Akbik, A., and Bross, J. 2009. Wanderlust: Extracting semantic relations from natural language text using dependency grammar patterns. In *Proceedings of the World Wide Web Conference (WWW2009) Semantic Search 2009 Workshop (SemSearch09)*.

Anderson, J. R. 2007. *How Can the Human Mind Occur in the Physical Universe?* Oxford and New York: Oxford University Press.

Bergen, B., and Chang, N. 2013. Embodied construction grammar. In Hoffman, T., and Trousdale, G., eds., *The Oxford Handbook of Construction Grammar*. New York: Oxford University Press. 168–190.

Bleiholder, J., and Naumann, F. 2008. Data fusion. *ACM Computing Surveys* 41(1):1–41.

Bryant, J. E. 2008. *Best-Fit Constructional Analysis*. Ph.D. Dissertation, University of California at Berkeley.

Buitelaar, P.; Cimiano, P.; Haase, P.; and Sintek, M. 2009. Towards linguistically grounded ontologies. In *Proceedings of the 6th European Semantic Web Conference (ESWC'09)*.

Chang, N. C.-L. 2008. *Constructing grammar: A computational model of the emergence of early constructions*. Ph.D. Dissertation, University of California at Berkeley.

Embley, D. W.; Kurtz, B. D.; and Woodfield, S. N. 1992. *Object-Oriented Systems Analysis: A Model-Driven Approach*. Englewood Cliffs, NJ: Yourdon Press.

Embley, D. W.; Liddle, S. W.; and Lonsdale, D. W. 2011. Conceptual modeling foundations for a web of knowledge. In Embley, D. W., and Thalheim, B., eds., *Handbook of Conceptual Modeling*. Springer. chapter 15.

Euzenat, J., and Schvaiko, P. 2007. *Ontology Matching*. Berlin: Springer.

Feldman, J. A. 2006. *From Molecule to Metaphor: A Neural Theory of Language*. Cambridge, MA: MIT Press.

Gruber, T. R. 1993. A translation approach to portable ontology specifications. *Knowledge Acquisition* 5(2):199–220.

Hoffman, T., and Trousdale, G., eds. 2013. *The Oxford Handbook of Construction Grammar*. New York: Oxford University Press.

Hruschka, E. R. J. 2013. Machine reading the web. Tutorial given at the 22nd International World Wide Web Conference, Rio de Janeiro, Brazil, 13-17 May, 2013.

Jackendoff, R. 1990. *Semantic Structures*. The MIT Press.

Jackendoff, R. 1996. Semantics and cognition. In Lappin, S., ed., *The Handbook of Contemporary Semantic Theory*. Blackwell.

Jackendoff, R. 2002. *Foundations of Language: Brain, Meaning, Grammar, Evolution*. Oxford University Press.

Jackendoff, R. 2003. Précis of foundations of language: Brain, meaning, grammar, evolution. *Behavioral and Brain Sciences* 26:651–707.

Johnson, M. 1987. *The Body in the Mind: The Bodily Basis of Meaning, Imagination, and Reason*. Chicago: The University of Chicago Press.

Laird, J. E. 2012. *The Soar Cognitive Architecture*. Cambridge, MA: The MIT Press.

Lakoff, G., and Johnson, M. 1980. The metaphorical structure of the human conceptual system. *Cognitive Science* 4:195–208.

Lewis, R. L. 1993. *An Architecturally-based Theory of Human Sentence Comprehension*. Ph.D. Dissertation, Carnegie Mellon University.

Lonsdale, D.; Hutchison, M.; Richards, T.; and Taysom, W. 2007. An NLP System for Extracting and Representing Knowledge from Abbreviated Text. In *Selected Proceedings of the Deseret Language and Linguistics Society Symposium*, 37–44. Brigham Young University.

Lonsdale, D.; Tustison, C.; Parker, C.; and Embley, D. 2008. Assessing clinical trial eligibility with logic expression queries. *Data & Knowledge Engineering* 66(1):3–17.

Melby, A. K., and Warner, C. T. 1995. *The Possibility of Language: A Discussion of the Nature of Language, with Implications for Human and Machine Translation*. Benjamin Translation Series. John Benjamins.

Mitra, P.; Noy, N. F.; and Jaizwal, A. R. 2004. Omen: A probabilistic ontology mapping tool. In *Proceedings of the Meaning Coordination and Negotiation workshop at the International Semantic Web Conference (ISwC)*, 537–547.

Mohan, S.; Mininger, A. H.; Kirk, J. R.; and Laird, J. E. 2012. Acquiring grounded representations of words with situated interactive instruction. *Advances in Cognitive Systems* 2:113–130.

Newell, A. 1990. *Unified Theories of Cognition*. Cambridge, MA: Harvard University Press.

Rytting, A., and Lonsdale, D. 2006. An operator-based account of semantic processing. In Lenci, A.; Montemagni, S.; and Pirrelli, V., eds., *Acquisition and representation of word meaning: theoretical and computational perspectives*. Pita/Rome: Istituti Editorialie Poligraphici Internazionali. 117–137.

Sarawagi, S. 2007. Information extraction. *Foundations and Trends in Databases* 1(3):261–377.

Sleator, D. D., and Temperley, D. 1993. Parsing English with a Link Grammar. In *Proceedings of the Third International Workshop on Parsing Technologies (IWPT)*.

Vanderpoel, G. B. 1902. *The Ely Ancestry: Lineage of Richard Ely of Plymouth England, who came to Boston, Mass., about 1655, & settled at Lyme, Conn, in 1660*. New York: The Calumet Press.